IN THE SPECIFICATION:

Please replace the paragraph beginning on line 24 of page 9 with the following:

Turning to FIG. 4, a flow diagram of the operation of one embodiment of software driver 150 of FIG. 2 is shown. Referring collectively to FIG. 2, FIG. 3A and FIG. 3B, the operation of software driver 150 of FIG. 2 is described. The flow diagram of FIG. 4 begins in step 400 where software driver 150 is in an idle state. If a new work request is made to software driver 150 or a work queue element completes, operation proceeds to step 410 where software driver 150 checks the request to see if the request is a new work request. If the request is not a new work request, operation proceeds to step 450 of FIG. 4 where software driver 150 checks a completion queue for the completed work queue element. Software driver 150 then clears the Busy bit 310 (e.g. the completion indicator) in FIG. 3~~A~~B for that corresponding work queue element. Operation then proceeds back to step 400 of FIG. 4. Referring back to step 410, if the request is a new work request, then operation proceeds to step 420. In step 420, software driver 150 checks for an available location in a queue pair such as QPx 210 of FIG. 3~~A~~B by checking the Busy bit 310 of each location in QPx 210. If there are no available locations, operation proceeds back to step 400 of FIG. 4 where software driver 150 may continue to try to deposit the new work queue element into QPx 210 at predetermined intervals. In step 420, if there are locations available such as location WQE_VA0, for example, then operation proceeds to step 430. In step 430, software driver 150 writes the work queue element into location WQE_VA0. Software driver 150 then sets the Busy bit 310, which indicates that this location is no longer available. Software driver 150 then writes the virtual address of the last location (e.g. WQE_VA0) in software register Last Work Queue Element Accessed 215 of FIG. 3~~A~~B. Software driver 150 then goes back to location WQE_VA0 and sets the Next WQE Posted Bit and writes the virtual address of the next available location (e.g. WQE_VA1) into the Next Virtual Address field of QPx 210. Operation then proceeds to step 440 of FIG. 4. Software driver 150 rings the hardware doorbell by writing to a corresponding QPx HW_Doorbell_Reg of FIG. 3~~B~~A, thus notifying hardware channel adapter 175 of a new work request. Operation then proceeds back to step 400.

Please replace the paragraph beginning on line 24 of page 10 with the following:

Referring now to FIG. 5, a flow diagram of the operation of one embodiment of hardware channel adapter 175 of FIG. 3A is shown. The operation of hardware channel adapter 175 begins in step 500 of FIG. 5. In step 500, hardware channel adapter 175 checks QPx HW_Doorbell_Reg of FIG. 3~~B~~A see if the count is not equal to zero. If the count is zero, then operation stays in step 500. However, if the count is not zero, then operation proceeds to step 510 where hardware channel adapter 175 checks to see if there is a new doorbell by checking to see if software driver 150 has written to QPx_Next_VA_Reg of FIG. 3A. If there is a new doorbell, operation proceeds to step 520 of FIG. 5, where hardware channel adapter 175 increments the HW_Doorbell_Reg. Operation then proceeds to step 530, where hardware channel adapter 175 checks to see if it is the first doorbell ring for that queue pair. The first doorbell ring notifies the hardware of the starting point of that particular queue pair in memory. Since queue pairs and work queue elements may be a predetermined size, the hardware may now track where each successive work queue element is located. If it is not the first doorbell ring, operation proceeds to step 550. Going back to step 530, if it is the first doorbell ring, operation proceeds to step 540, where hardware channel adapter 175 copies data written to QPx_Next_VA_Reg of FIG. 3~~B~~A into QPx_Next_VA_Reg and operation proceeds to step 550. Going back to step 510, if there is no new doorbell and the HW_Doorbell_Reg of FIG. 3~~B~~A is not zero, operation proceeds to step 550 of FIG. 5. In step 550, hardware channel adapter 175 checks if necessary resources are available to service a work queue element. If the resources are not available, operation proceeds back to step 500. If the resources are available, operation proceeds to step 560 where hardware channel adapter 175 fetches the virtual address in QPx_Next_VA_Reg of FIG. 3~~B~~A and reads the work queue element stored there. Operation then proceeds to step 570. Hardware channel adapter 175 then checks the In_Service_Bit of FIG. 3A. If the In_Service_Bit was not set, operation proceeds to step 580. In step 580, hardware channel adapter 175 places the work queue element into service, sets the In_Service_Bit of FIG. 3A and decrements the HW_Doorbell_Reg. Operation then proceeds back to step 500 of FIG. 5. Referring back to step 570, if the In_Service_Bit of FIG. 3A is set, then operation proceeds to step 590 of FIG. 5. In step 590, hardware channel adapter 175 clears the Next_WQE_Posted bit of FIG. 3B and loads the Next Virtual Address into QPx_Next_VA_Reg of FIG. 3A. Operation then proceeds back to step 500.